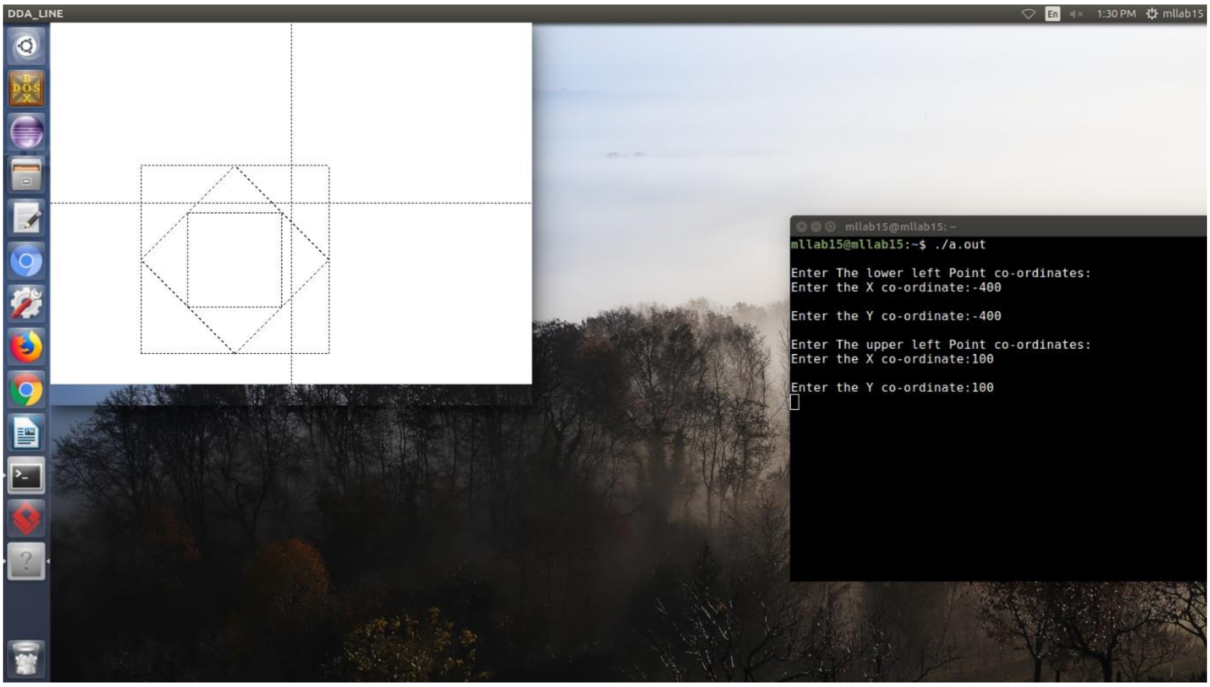
Assignment-2

```
#include<stdio.h>
#include<GL/freeglut.h>
#include<GL/glut.h>
#include<math.h>
int X1,X2,Y1,Y2;
void print_point(int x,int y)
{
        glBegin(GL_POINTS);
        glVertex2i(x,y);
        glEnd();
}
int round_off(int v)
{
 return floor(v + 0.5);
}
void dda(float x1,float y1,float x2,float y2)
{
        float xinc,yinc;
        float x=x1,y=y1;
        float dx = (x2-x1);
        float dy = (y2-y1);
        int steps = abs(dx)>abs(dy)?dx:dy;
        xinc = dx/(float)steps;
        yinc = dy/(float)steps;
        print_point(x,y);
        for(int i=0;i<steps;i++)
        {
                x+=xinc;
                y+=yinc;
                if(i%10<5)
                print_point(round_off(x),round_off(y));
}
}
void draw()
{
        dda(-640,0,640,0);
        dda(0,-480,0,480);
        dda(X1,Y1,X2,Y1);
        dda(X2,Y1,X2,Y2);
        dda(X1,Y2,X2,Y2);
        dda(X1,Y1,X1,Y2);
        dda((X2+X1)/2,Y1,X2,(Y2+Y1)/2);
        dda(X2,(Y2+Y1)/2,(X2+X1)/2,Y2);
        dda(X1,(Y2+Y1)/2,(X2+X1)/2,Y2);
        dda((X2+X1)/2,Y1,X1,(Y2+Y1)/2);
        dda((X2+3*X1)/4,(Y2+3*Y1)/4,(3*X2+X1)/4,(Y2+3*Y1)/4);
        dda((3*X2+X1)/4,(Y2+3*Y1)/4,(3*X2+X1)/4,(3*Y2+Y1)/4);
        dda((X2+3*X1)/4,(3*Y2+Y1)/4,(3*X2+X1)/4,(3*Y2+Y1)/4);
        dda((X2+3*X1)/4,(Y2+3*Y1)/4,(X2+3*X1)/4,(3*Y2+Y1)/4);
}
void init()
{
        glClearColor(1.0,1.0,1.0,0.0);
        glClear(GL_COLOR_BUFFER_BIT);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glColor3f(0.0,0.0,0.0);
        gluOrtho2D(-640,640,-480,480);
        draw();
}
```

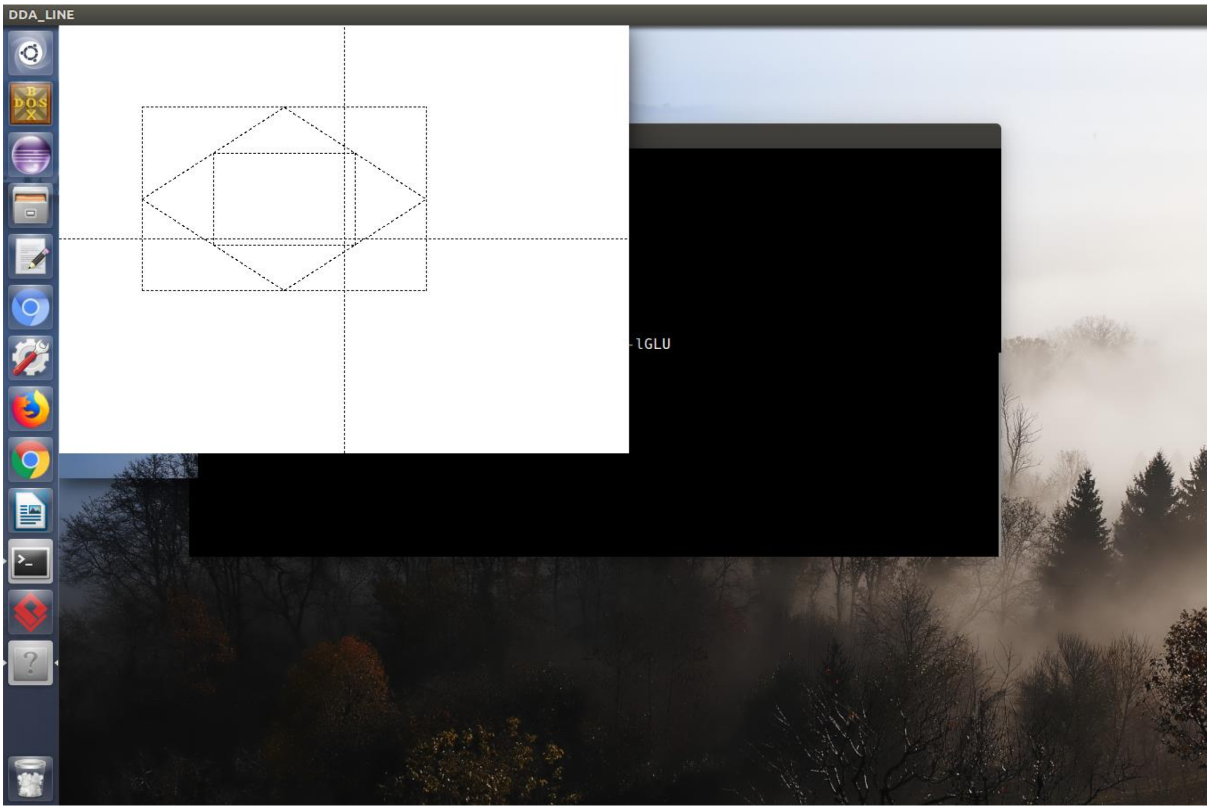
```
int main(int argc,char **argv)
{
	printf("\nEnter The lower left Point co-ordinates:");
	printf("\nEnter the X co-ordinate:");
	scanf("%d",&X1);
	printf("\nEnter the Y co-ordinate:");
	scanf("%d",&Y1);
	printf("\nEnter The upper left Point co-ordinates:");
	printf("\nEnter the X co-ordinate:");
	scanf("%d",&X2);
	printf("\nEnter the Y co-ordinate:");
	scanf("%d",&Y2);
	glutInit(&argc,argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(640,480);
	glutInitWindowPosition(0,0);
	glutCreateWindow("DDA_LINE");
	init();
	glutDisplayFunc(draw);
	glutMainLoop();
	return 0;
}
```



```
mllab15@mllab15:~$ ./a.out

Enter The lower left Point co-ordinates:
Enter the X co-ordinate:-400

Enter the Y co-ordinate:-400

Enter The upper left Point co-ordinates:
Enter the X co-ordinate:100

Enter the Y co-ordinate:100
```

```
#include<stdio.h>
#include<GL/freeglut.h>
#include<GL/glut.h>
#include<math.h>
int X1,X2,Y1,Y2;
void print_point(int x,int y)
{
glBegin(GL_POINTS);
glVertex2i(x,y);
glEnd();
}
int round_off(int v)
{
 return floor(v + 0.5);
}
void dda(float x1,float y1,float x2,float y2)
{
float dx = (x2-x1);
float dy = (y2-y1);
int steps = abs(dx)>abs(dy)?dx:dy;
float xinc,yinc;
float x=x1,y=y1;
xinc = dx/(float)steps;
yinc = dy/(float)steps;
print_point(x,y);
for(int i=0;i<steps;i++)
{
x+=xinc;
y+=yinc;
if(i%10<5)
print_point(round_off(x),round_off(y));
}
}
void draw()
{
dda(X1,Y1,X2,Y1);
dda(X2,Y1,X2,Y2);
dda(X1,Y2,X2,Y2);
dda(X1,Y1,X1,Y2);
dda((X2+X1)/2,Y1,X2,(Y2+Y1)/2);
dda((X2+X1)/2,Y2,X2,(Y2+Y1)/2);
dda(X1,(Y2+Y1)/2,(X2+X1)/2,Y2);
dda(X1,(Y2+Y1)/2,(X2+X1)/2,Y1);
dda((X2+3*X1)/4,(Y2+3*Y1)/4,(3*X2+X1)/4,(Y2+3*Y1)/4);
dda((3*X2+X1)/4,(Y2+3*Y1)/4,(3*X2+X1)/4,(3*Y2+Y1)/4);
dda((X2+3*X1)/4,(3*Y2+Y1)/4,(3*X2+X1)/4,(3*Y2+Y1)/4);
dda((X2+3*X1)/4,(Y2+3*Y1)/4,(X2+3*X1)/4,(3*Y2+Y1)/4);
}
void init()
{
glClearColor(1.0,1.0,1.0,0.0);
glClear(GL_COLOR_BUFFER_BIT);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glColor3f(0.0,0.0,0.0);
gluOrtho2D(-640,640,-480,480);
dda(-640,0,640,0);
dda(0,-480,0,480);
}

void mouse(int button,int state,int x,int y)
```

```
{
if(button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
{
X1 = 2*x - 640;
Y1 = 480 - 2*y ;
}
else if(button == GLUT_LEFT_BUTTON && state == GLUT_UP)
{
X2 = 2*x - 640;
Y2 = 480 - 2*y;
draw();
}
}
int main(int argc,char **argv)
{
glutInit(&argc,argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(640,480);
glutInitWindowPosition(0,0);
glutCreateWindow("DDA_LINE");
init();
glutMouseFunc(mouse);
glutDisplayFunc(draw);
glutMainLoop();
return 0;
}
```



```
-lGLU
```